6.18 Meeting Records

- Discussed Yue's analysis of ChalametPIR
  - Dimension of the database. Illustration:https://raw.githubusercontent.com/helloboyxxx/images-for-notes/master/uPic/image-20240616152519435.png
  - A few details of BFF. Answered a few questions: Is longer key length going to work better? How does the key length affect false-positive rate? Why use $k \in \{3, 4\}$ many hash functions for hashing? Why not more hash functions?
- Can cuckoo hashing method work better than ChalametPIR?
  - Even if we double the database size for storing the hash table, and make two queries for each keyword, it could be better than 4 queries on the database that has 1.13 blowup. One thing to figure out: what is the false-positive rate in BFF used by ChalametPIR? Yue said it was 2%, but is it?
- What are other methods:
  - Constant-weight Equality Operators is not good enough. Not suitable in our settings.
  - SparsePIR could work, but we don't have their code.

TODO

- Figure out the false-positive rate.

- Evaluate if SparsePIR is better than the naive **cuckoo hashing method**. Sparse PIR didn't share their code along with their paper, but prof. Ren will try to contact them and get the code :). For a better comparison / evaluation, I think it is better to write down a decent description of the **cuckoo hashing method** we have been refering to. Prof. Ren mentioned that SparsePIR do have some drawback. We need to figure that out.
- Figure out if there is indeed a bug in ChalametPIR's F.setup pseudocode.

Food for thought

- Simple and Practical Amortized Sublinear Private Information Retrieval. This is the State-of-art stateful PIR. Worth reading.
- Suppose the value we store inside PIR has very different length (say, wiki pages), can we develop a better scheme instead of padding every value to the max length?